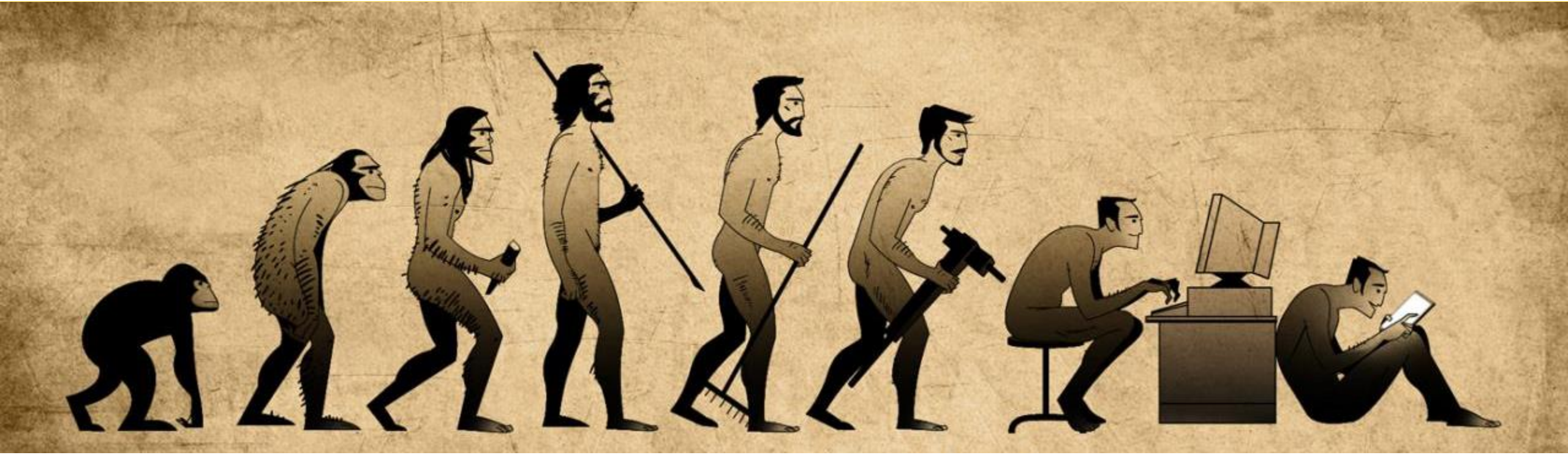
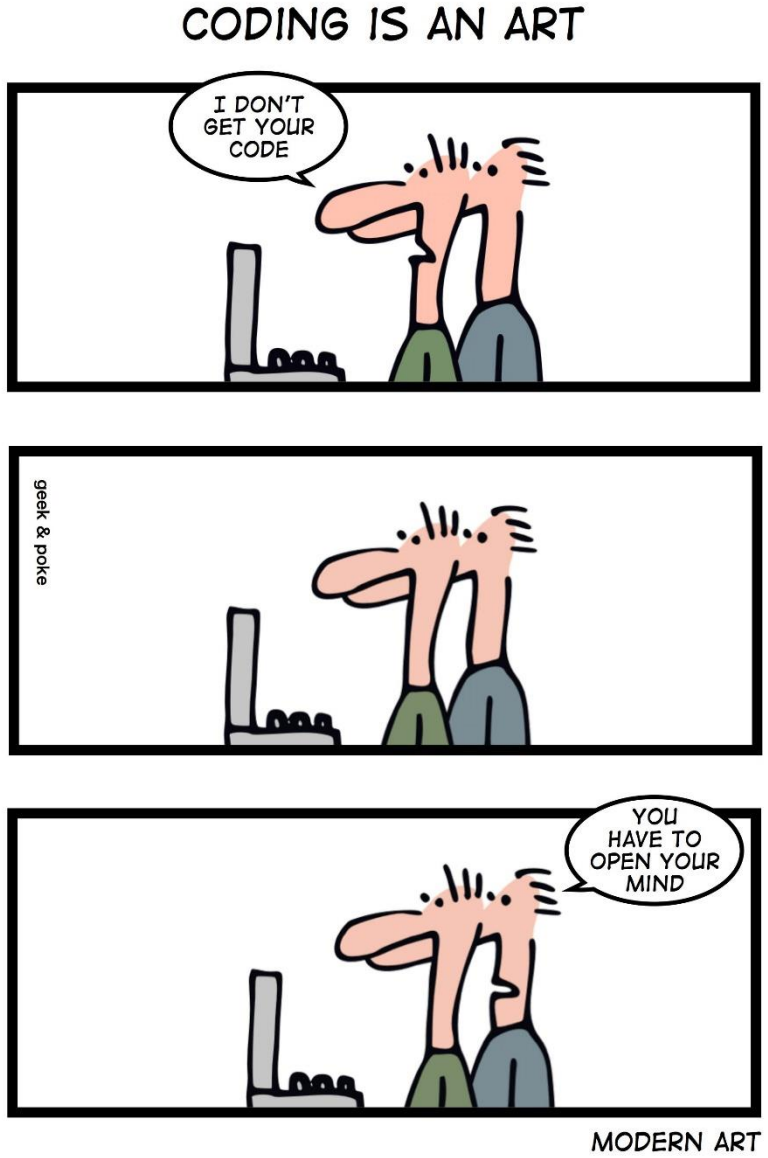
Introduction à JavaScript



Antoine SCHMID, CFPT-Informatique
Avec la collaboration du CPLN

Table des Matières

- Introduction à la programmation
- Histoire de JavaScript
- Un premier programme
- Où placer son code JavaScript



Informatique

- Depuis son invention dans les années 1950, l'**informatique** a révolutionné bien des domaines de notre vie quotidienne.
  - Le calcul d'un itinéraire depuis un site Internet ou un GPS
  - La réservation à distance d'un billet de train ou d'avion
  - La possibilité de voir et de parler avec des amis à l'autre bout du monde

- Mais un ordinateur, même très performant, n'est qu'une machine capable d'exécuter **automatiquement** une série d'opérations simples qu'on lui a demandées.

- L'intérêt des ordinateurs est de savoir manipuler très rapidement et sans erreur d'énormes quantités d'informations

- **Informatique = INFORMA**tion auto**MATIQUE**

Programmeur / programme

- Le rôle du **programmeur** (appelé également **développeur**) est de fournir les ordres que la machine doit exécuter en écrivant des programmes.

- Un **programme informatique** (également appelé **application** ou **logiciel**) est une liste d'ordres indiquant à un ordinateur ce qu'il doit faire.
  - Un ou (le plus souvent) plusieurs fichiers contenant les ordres donnés à la machine, qu'on appelle également des **instructions**.
  - L'ensemble des fichiers contenant les instructions du programme constitue son **code source**.
  - Programmer, c'est donc écrire le code source d'un programme, d'où l'emploi du terme **coder**.

Assembleur

- Le seul langage de programmation directement compréhensible par un ordinateur est le **langage machine**, également appelé **assembleur**.
  - Lié a un type de processeur
  - Permet de manipuler directement la mémoire de la machine

```
1  str:
2   .ascii "Bonjour\n"
3   .global _start
4
5  _start:
6  movl $4, %eax
7  movl $1, %ebx
8  movl $str, %ecx
9  movl $8, %edx
10 int $0x80
11 movl $1, %eax
12 movl $0, %ebx
13 int $0x80
```

Les langages de programmation

- Il existe un grand nombre de **langages de programmation**, adaptés à des usages variés.
- Chaque langage de programmation dispose de sa propre syntaxe et d'instructions spécifiques
- Cela dit, on peut dégager des similitudes entre les langages de programmation les plus courants

python

```python
print("Bonjour")
```

php

```php
<?php
echo("Bonjour\n");
?>
```

Les langages de programmation

csharp

```csharp
class Program {
    static void Main(string[] args) {
        Console.WriteLine("Bonjour");
    }
}
```

java

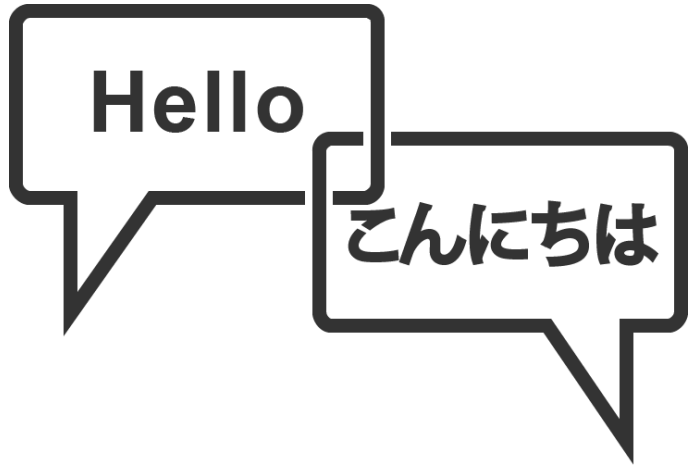
```java
public class Program {
    public static void main(String[] args) {
        System.out.println("Bonjour");
    }
}
```

- Tous ces programmes affichent le message "Bonjour", mais chacun d'eux le fait à sa manière.

Exécution d'un programme

- On nomme **exécution** le fait de demander à un ordinateur de réaliser les ordres contenus dans un programme.

- Quel que soit le langage avec lequel il est écrit, un programme doit être traduit en assembleur pour pouvoir être exécuté.



- Ce processus de traduction dépend du langage choisi.

Interpréteur

- Un **interpréteur** est un programme informatique qui traite le code source d'un projet logiciel pendant son fonctionnement
  - Procède toujours ligne de code par ligne de code
  - Les différentes instructions sont lues, analysées et préparées pour le processeur dans l'ordre

- Parmi les langages de programmation les plus célèbres ayant majoritairement recours à un interpréteur pour la conversion du code source en code machine, on compte notamment **BASIC**, **Perl**, **Ruby** et **PHP**.
- Ces langages sont d'ailleurs souvent réunis sous le terme de **langages interprétés**.

Compilateur

- Un **compilateur** est un programme informatique qui traduit l'ensemble du code source d'un projet logiciel en code machine avant son exécution

- Parmi les langages entièrement **compilés**, on compte notamment des piliers tels que **C**, **C++** et **Pascal**.

Solutions hybrides

- Un **pseudo-compilateur** est un programme informatique qui traduit l'ensemble du code source en **pseudo-code**.
- Le pseudo-code est ensuite interprété par une **machine virtuelle** au moment de l'exécution
- Le programme ainsi généré peut être exécuté sur sur n'importe quelle plate-forme supportant **l'environnement**.

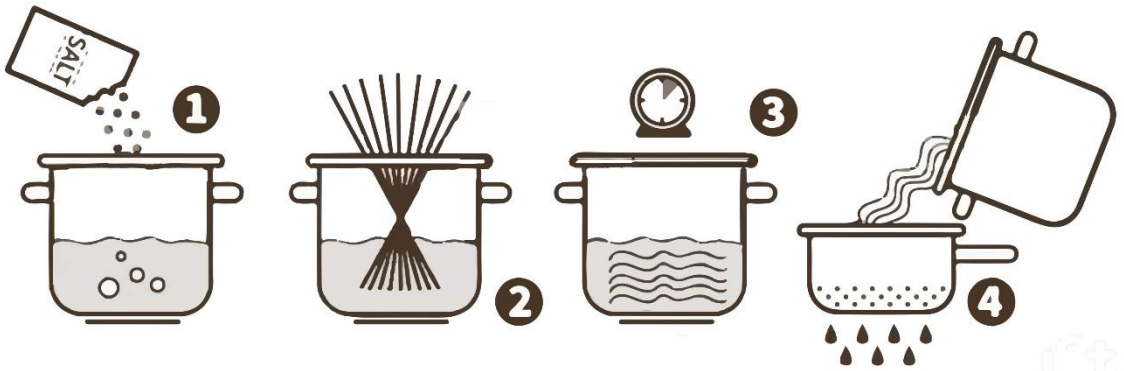
- C'est le cas du langage **Java** et des langages de la plate-forme Microsoft **.NET** (**C#, VB.NET**)

Programmer

- Sauf dans des cas très simples, on ne crée pas un programme en se lançant directement dans l'écriture du code source.
- Il est d'abord nécessaire d'analyser le problème pour trouver la suite d'opérations à réaliser pour le résoudre.

- Je souhaite me préparer un plat de pâtes. Quelles sont les étapes qui vont me permettre d'atteindre mon objectif ? (à vos crayons)

Programmer




```
Début
    Sortir une casserole
    Mettre de l'eau dans la casserole
    Ajouter du sel
    Mettre la casserole sur le feu
    Tant que l'eau ne bout pas
      Attendre
    Sortir les pâtes du placard
    Verser les pâtes dans la casserole
    Tant que les pâtes ne sont pas cuites
        Attendre
    Verser les pâtes dans une passoire
    Egoutter les pâtes
    Verser les pâtes dans un plat
    Goûter
    Tant que les pâtes sont trop fades
        Ajouter du sel
        Goûter
    Si on préfère le beurre à l'huile
        Ajouter du beurre
    Sinon
        Ajouter de l'huile
Fin
```

Algorithme

- On peut distinguer différents types d'actions :
  - des actions simples ("Sortir une casserole") ;
  - des actions conditionnelles ("Si on préfère le beurre à l'huile...") ;
  - des actions qui se répètent ("Tant que les pâtes sont trop fades...").

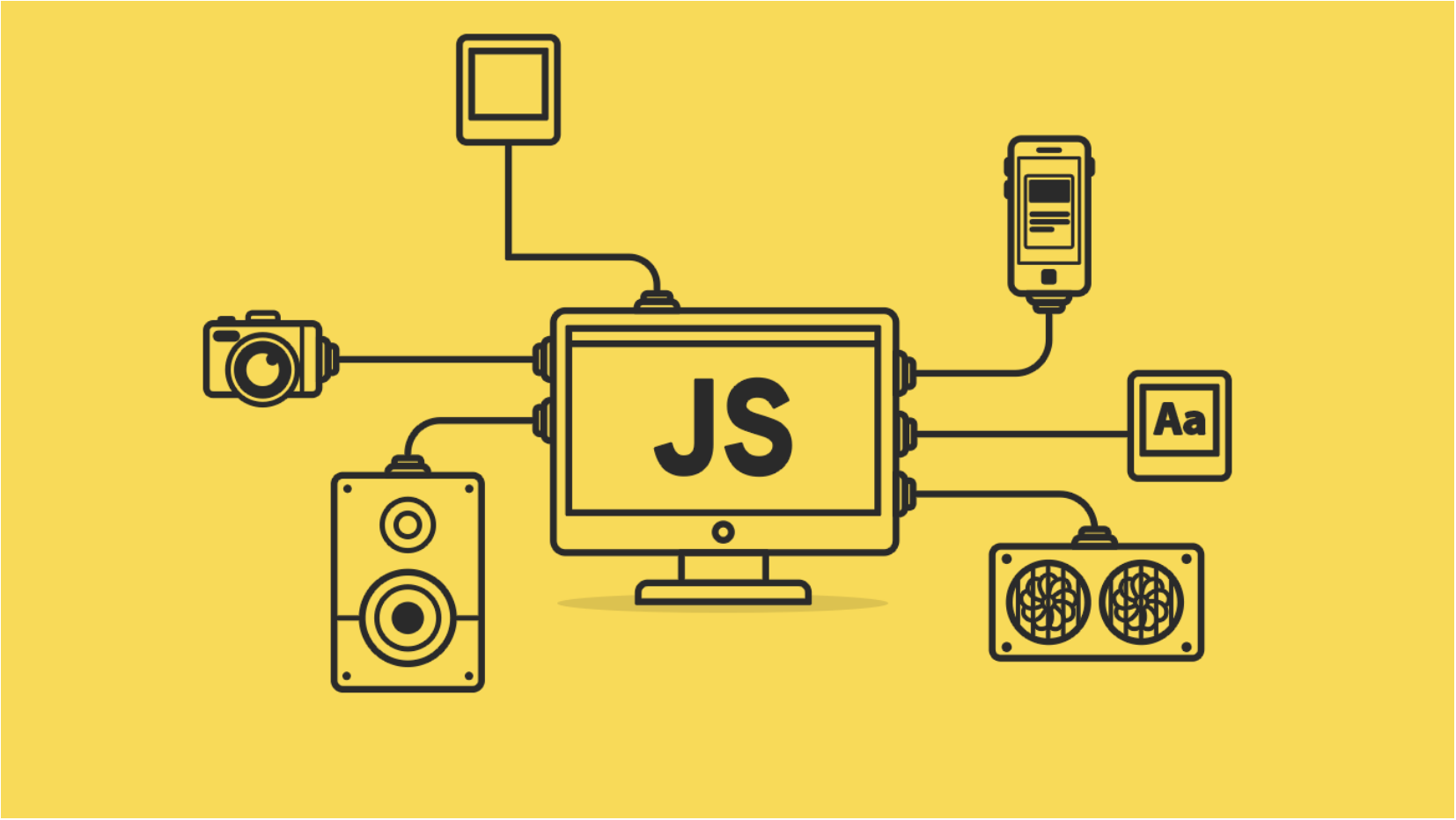
- Un **algorithme** est **une suite ordonnée d'opérations permettant de résoudre un problème donné**.
- Un algorithme décompose un problème complexe en une suite d'opérations simples.

Le programmeur

- Écrire des programmes qui réalisent de manière fiable les tâches attendues est la première mission du programmeur.
- Un débutant arrivera vite à créer des programmes simples. La difficulté apparaît lorsque que le programme évolue et se complexifie. Il faut de l'expérience et beaucoup de pratique avant d'arriver à maîtriser cette complexité.

"Le programmeur est un créateur d'univers dont il est seul responsable.
Des univers d'une complexité virtuellement infinie peuvent
être créés sous la forme de programmes informatiques."
(Joseph Weizenbaum)

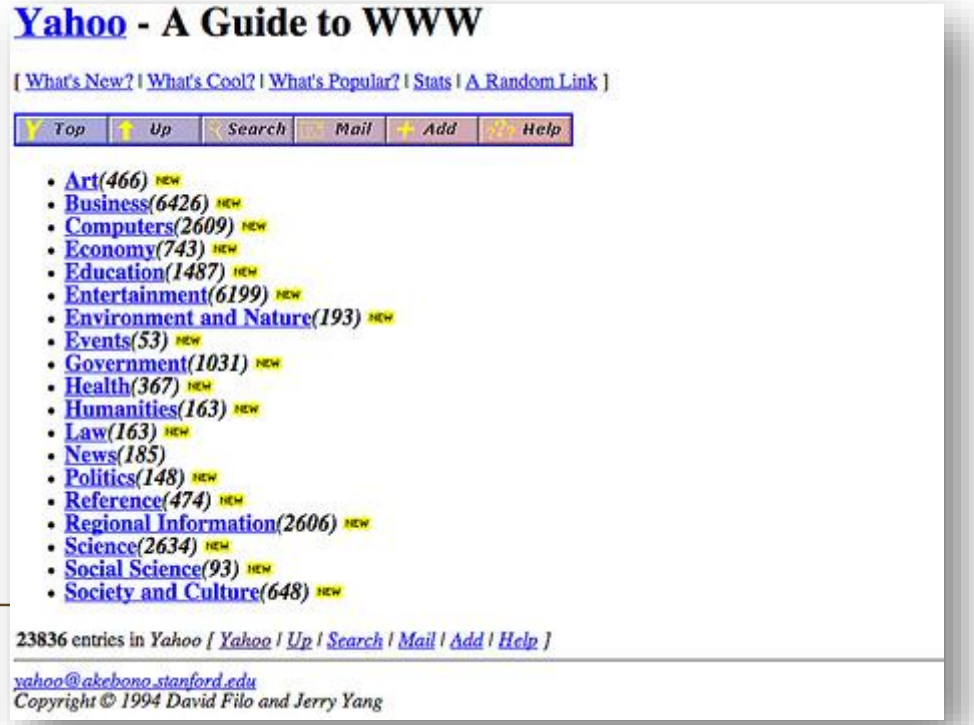
Histoire de JavaScript



Origine

- JavaScript est avant tout le langage de programmation du Web.
- Il a été inventé en 1995 par Brendan Eich, qui travaillait à l'époque pour la société Netscape, créatrice du premier navigateur Web populaire (l'ancêtre de Firefox).

- L'idée de départ était de créer un langage simple pour rendre dynamiques et interactives les pages Web,
qui étaient très simplistes à l'époque.



Origine

- Petit à petit, les créateurs de sites Web ont enrichi leurs pages en y ajoutant du code écrit en JavaScript.
- Pour que le résultat fonctionne, il fallait que le navigateur Web comprenne le JavaScript.
- Ce langage a donc été progressivement intégré à l'ensemble des navigateurs.
- N'importe quel navigateur Web est aujourd'hui capable d'exécuter du code écrit en JavaScript.

Origine

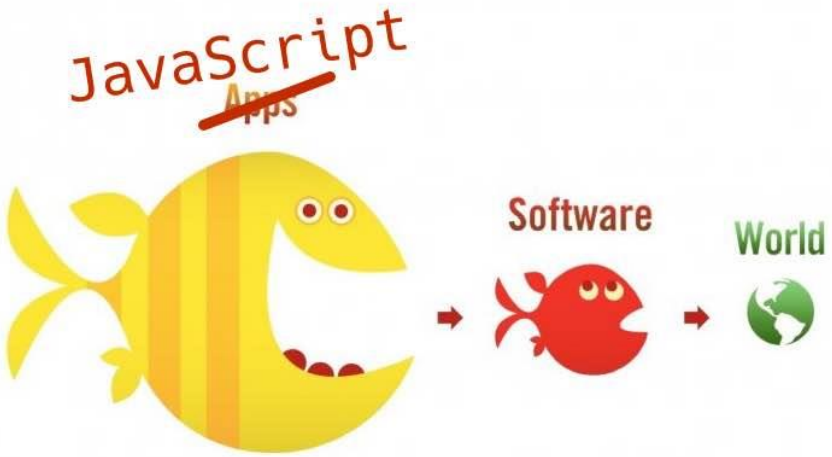
- L'explosion du Web puis l'avènement du Web 2.0, basés sur des pages riches et interactives, ont rendu JavaScript de plus en plus populaire.
- Les concepteurs de navigateurs Web ont optimisé la rapidité d'exécution du code JavaScript, jusqu'à en faire un langage très performant.
- Cela a conduit à l'apparition en 2009 de la plate-forme Node.js, qui permet d'écrire en JavaScript des applications Web très rapides.

Origine

- Enfin, l'arrivée des smartphones et autres tablettes dotés de systèmes différents et incompatibles (iOS, Android ou Windows Phone) a conduit à l'apparition d'outils de développement dits multiplateformes.
- Ils permettent d'écrire en une seule fois des applications mobiles compatibles avec l'ensemble des terminaux du marché.
- Ces outils sont presque toujours basés sur... JavaScript !

Origine

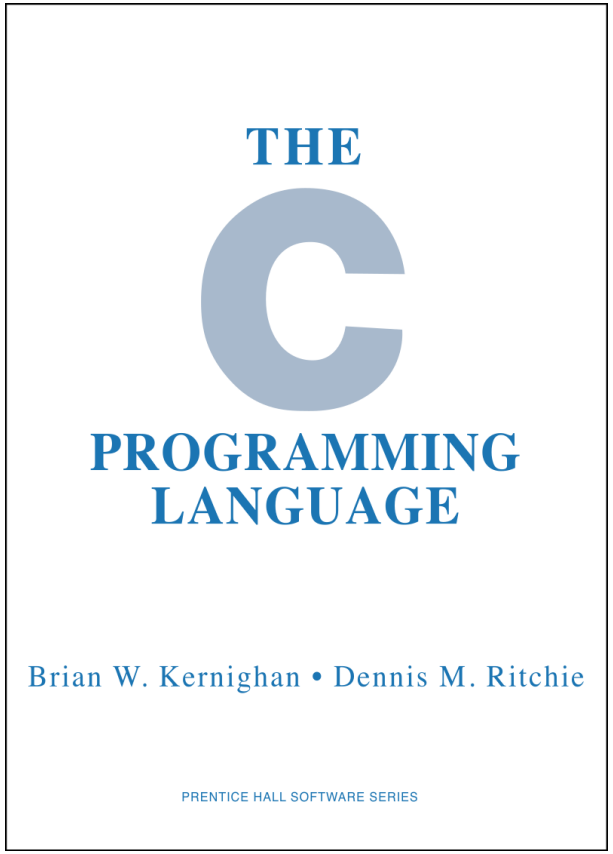
- Ce petit résumé de l'histoire de JavaScript montre à quel point ce langage est maintenant présent partout. Il dispose d’un foisonnant écosystème de composants (des "briques de base" qu'on peut intégrer pour construire un logiciel) et d’une immense communauté de développeurs.



Un premier programme

Origine

- C’est par tradition le premier programme qu’on fait dans chaque langage de programmation
- Première version en 1972 (B)
- Dans la littérature dès 1978



JUST DO IT.

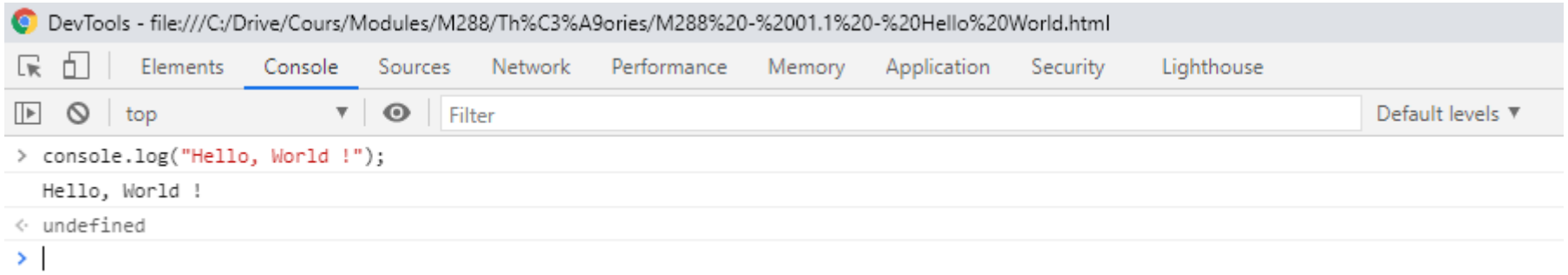
Hello, World !

```
console.log("Hello, World !");
```

- Comment exécuter et tester votre code ?

Environnement – Première solution

- Taper votre code directement dans la console JavaScript de votre navigateur web
- Basculez dans le mode «Outils de développement» de votre navigateur (F12)
- Cliquez sur l'onglet «Console»



Environnement – Deuxième solution

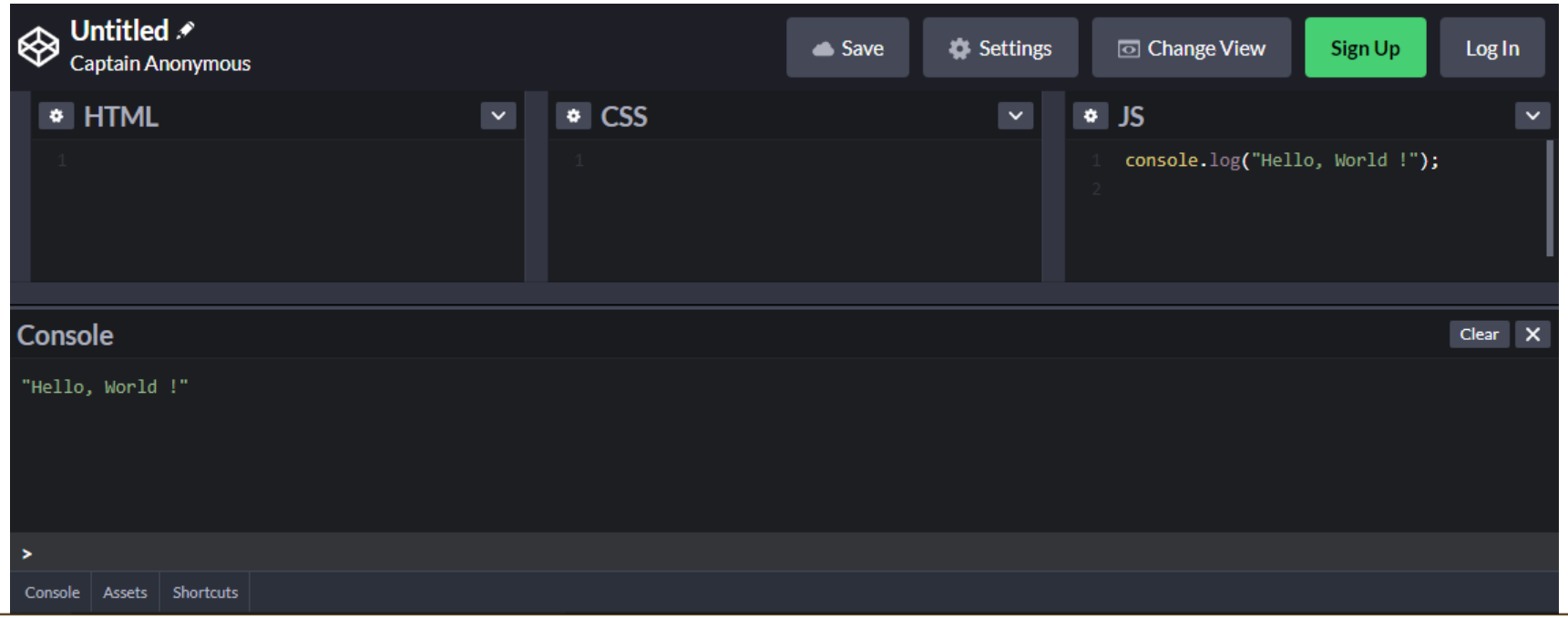
- Utilisez un éditeur de texte (Visual Studio Code par exemple) et créez une page web vide
- Insérez votre code JavaScript entre les balises `<script></script>`
- Regardez dans la console de votre navigateur pour voir le résultat

```
<!DOCTYPE html>
<html>
  <head>
    <!-- en-tete du document -->
    <title>Hello World</title>
  </head>

  <body>
    <script type="text/javascript">
      console.log("Hello, World !");
    </script>
  </body>
</html>
```

Environnement – Troisième solution

- Il existe des environnement web gratuit intéressants
- Ex: https://codepen.io/



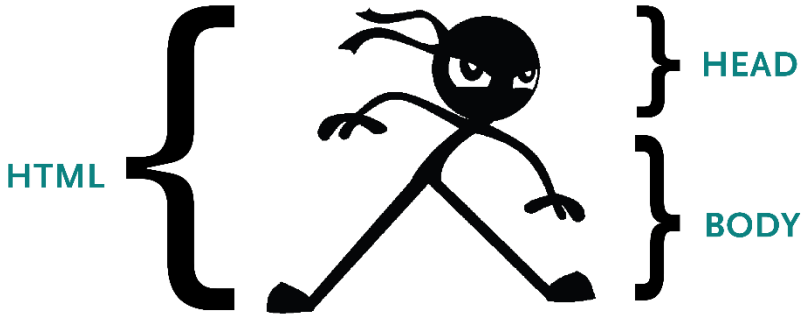
Où placer son code JavaScript ?

Les balises

- Le JavaScript s'écrit entre des balises `<script></script>`
- Pour indiquer à votre navigateur que c'est du JavaScript, on ajoute la propriété `type="text/javascript"`

```
<script type="text/javascript">
    /* votre code se trouve ici
       et peut avoir autant de lignes
       que nécessaire */
</script>
```

En-tête vs corps



- Dans le corps de la page <body>…</body>
  - les scripts à exécuter au chargement de cette dernière (lorsque le navigateur "lira" le code, il l'exécutera en même temps).
  - il suffit d'écrire le code à exécuter entre les balises

- Dans l'en-tête <head>…</head>
  - les éléments qui doivent être exécutés plus tard (lors d'un événement particulier, par exemple).
  - Dans ce cas, le code n'est pas écrit "en vrac", nous apprendrons plus loin comment l'organiser.

Exercice

- Dans une page web, on veut:
  - une boîte de dialogue indiquant le début du chargement de la page (donc, le code est à placer au début du corps de la page),
  - une autre indiquant la fin du chargement de celle-ci (donc, à la fin du corps).

- Exemple de boîte de dialogue

```
alert("Oh la jolie boîte !");
```

```
<!DOCTYPE html>
<html>
<head>
  <!-- en-tete du document -->
  <title>Un exemple</title>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
</head>
<body>
  <!-- script pour le début du chargement -->
  <script type="text/javascript">
    alert("Debut du chargement de la page");
  </script>
  <!-- ici se trouve le contenu de la page web -->
  <p>
    Vous testez un script...<br />
    Enjoy ;)
  </p>
  <!-- script pour la fin du chargement -->
  <script type="text/javascript">
    alert("Fin du chargement de la page");
  </script>
</body>
</html>
```

Constatations

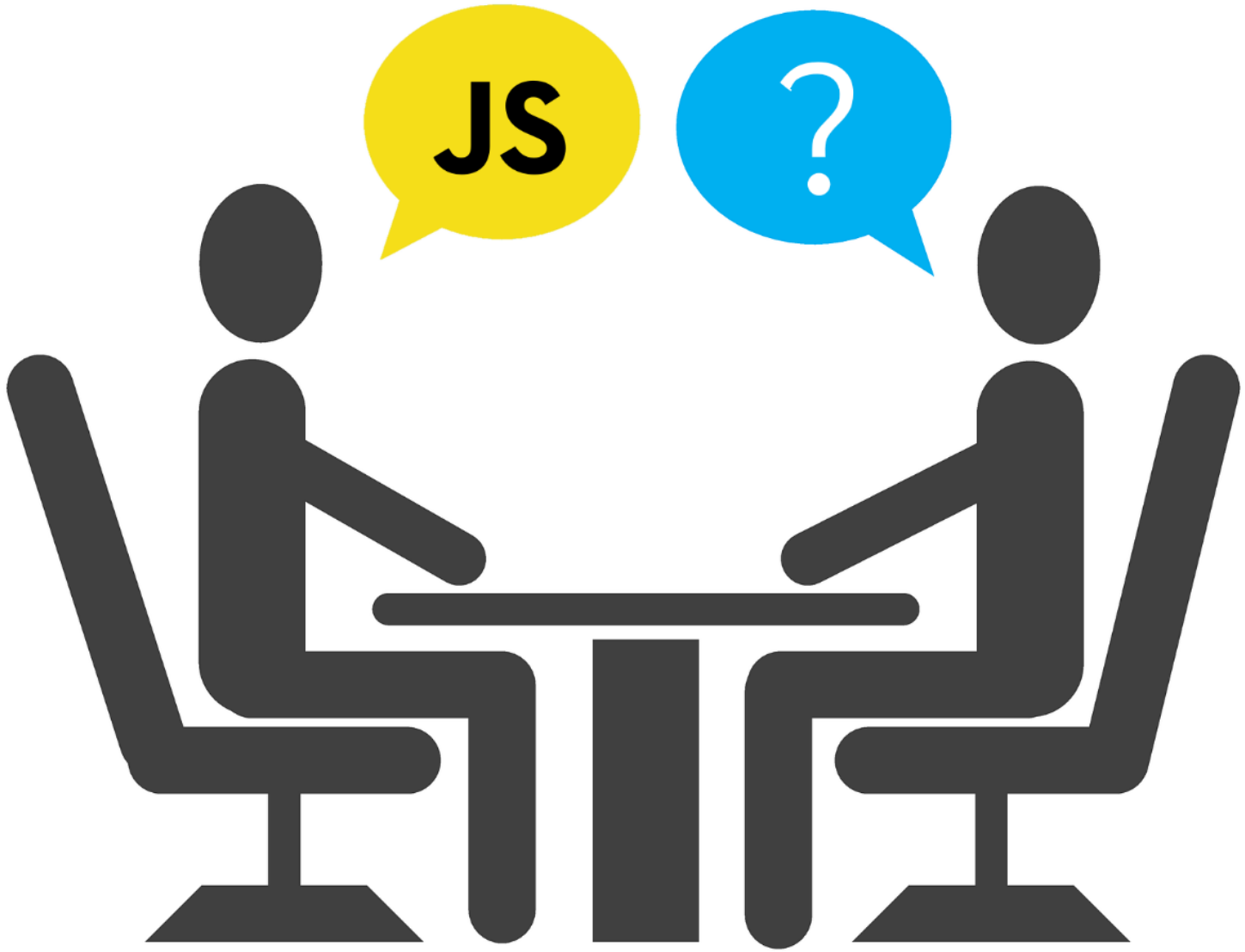
- Vous remarquez que tant que la première boîte de dialogue est ouverte, la page n'est pas chargée.
- En effet, le navigateur exécute le JS au fur et à mesure du chargement de la page
- Il attend donc que le script soit terminé avant de charger la suite.

Importer un fichier externe

- Il est possible de placer le code dans un fichier indépendant
- On dit que le code est **importé** depuis un fichier externe.
- L'extension d'un fichier externe contenant du code JavaScript est **.js**

```
<script type="text/javascript" src="script.js"></script>
```

- On va indiquer aux balises le nom et l'emplacement du fichier contenant notre script, grâce à la propriété `src` (comme pour les images).


JS
?